Attorney's Docket No.: 06618/303001 / CIT2810

REMARKS

Reconsideration and allowance of the above referenced application are respectfully requested. Claims 1-6 remain pending.

Claims 1-4 standard rejected under 35 USC 102b as allegedly being anticipated by U.S. patent No. 5,584,2035 to Duggin. This contention has been obviated by the amendment of claim 1 to emphasize its patentable distinctions. As amended, it is respectfully suggested that all of the claims should be in condition for allowance.

Specifically, the present system describes a special kind of distributed system in which each of the specified process objects of the system is capable of becoming any other kind of process object. That is, this distributed system, unlike Duggin's system, does not have designated processes that carry out specialized tasks. Rather, each of the plurality of process objects in the present system is able to create an inbox, an outbox, and interconnect the inbox and outbox by passing messages therebetween.

This is very different than Duggin who passes all of his messages through the object manager (OM) local to the object transmitting the message. This system connects each created outbox to a created inbox, thereby establishing a personal network, as defined by claim 1. The system needs no object

manager or any other manager process or manager object. Rather, this system uses a dynamic collection of substantially equivalent processes. Therefore, it is respectfully suggested that this system is very different than Duggins.

In addition, claim 1 defines a special freeze method that freezes the state of the process object to a persistent storage to convert that object into one that does not use system resources. The object can then be thawed for use. Admittedly, Duggin does teach saving the state of the process object as explained col. 12 line 43-45. This object is saved to disk when it receives a quit message. The object is saved to disk and execution is suspended. However, in the present system, the process object can be changed into a frozen object while the rest of the objects are operating. This allows the system to continue execution. It does not need to change or suspend execution or change its behavior in any way. This difference may be important since the object states are saved as the system executes. In contrast, Duggins only enables the system to quit; it does not being enable freezing a state of process object. Therefore, this system would not be able to allocate resources for as well as the current system.

Therefore, claim 1 should be allowable along with the claims which depend therefrom.

Attorney's Docket No.: 06618/303001 / CIT2810

In view of the above amendments and remarks, all of the claims should be in condition for allowance. A formal notice to the effect is respectfully solicited.

Please apply any other charges or credits to Deposit Account No. 06-1050.

Respectfully submitted,

Date: 10/3/02

Scott C. Harris
Reg. No. 32,030

Fish & Richardson P.C.
Customer Number: 20985 * [redacted] *
4350 La Jolla Village Drive, Suite 500
San Diego, California 92122
Telephone: (858) 678-5070
Facsimile: (858) 678-5099
10219471.doc

Attached is a marked-up version of the changes being made by the current amendment.

Version with markings to show changes made

In the claims:

Please amend the claims as follows:

1. (Amended) A distributed system framework for a networked environment, including a plurality of process objects, each of said process objects including:

(a) a program method for creating at least one inbox for storing messages received from another process object;

(b) a program method for creating at least one outbox for storing messages to be transmitted to another process object;

(c) a freeze method that saves the state of the process object to persistent storage, thereby changing the process object to a frozen process object which does not use operating system resources;

(d) a thaw method that restores the frozen process object from the persistent storage, thereby changing the frozen process object to a ready process object;

(e) a program method for interconnecting each created outbox of the process object to a created inbox of at least one other process object, thereby establishing a personal network between the process object and such other process objects within a communication session to perform at least

one task by passing messages between the interconnected outboxes and inboxes.

2. The distributed system framework of claim 1, each process object further including at least one mail daemon object, for controlling the order of messages in each inbox.

3. The distributed system framework of claim 1, each process object further including a communing response method, for instantiating the process object if the process object is summoned by another process object.

4. The distributed system framework of claim 3, wherein the summoning response method causes the thaw method of the process object to be invoked if the process object is frozen when summoned by another process.

5. The distributed system framework of claim 1, wherein each message includes a snapshot variable that indicates whether a process object has recorded its state.

6. The distributed system framework of claim 5, wherein the snapshot variable is a date field, and each process object includes a snapshot method that saves the state of the process object if a date field value within a received message is later that an current date value for the process object, updates the current date value for the process object to the date field value of the received message, and increments a clock for the process object to a value exceeding the date field value of the received message.